CMPSC 465 Data Structures & Algorithms
Spring 2024 Mehrdad Mahdavi and David Koslicki

HW 2

Due January 25, 10:00 pm

Instructions: You are encouraged to solve the problem sets on your own, or in groups of up to five people, but you must write your solutions strictly by yourself. You must explicitly acknowledge in your write-up all your collaborators, as well as any books, papers, web pages, etc. you got ideas from.

Formatting: Each problem should begin on a new page. Each page should be clearly labeled with the problem number. The pages of your homework submissions must be in order. You risk receiving no credit for it if you do not adhere to these guidelines.

Late homework will not be accepted. Please, do not ask for extensions since we will provide solutions shortly after the due date. Remember that we will drop your lowest two scores.

This homework is due Thursday, January 25, at 10:00 pm electronically. You need to submit it via Gradescope (Class code XX7RVV). Please ask on Campuswire about any details concerning Gradescope and formatting.

1. (20 pts.) **Big-O notation.** Assume you have functions f and g such that $f(n)$ is $O(g(n))$. For each of the following statements, decide whether it is true or false. Give a proof for the true statements and a counterexample for the false ones. (Hint: Think carefully before committing to an answer. Sometimes the answer that seems "obvious" might be wrong.)

(a) $2^{f(n)}$ is $O(2^{g(n)})$
(b) $f(n)^2$ is $O(g(n)^2)$
(c) $2^{f(n)}$ is $O(2^{O(g(n))})$
(d) $\log_2 f(n)$ is $O(\log_2 g(n))$ where $g(n) > 1$

2. (20 pts.) **Polynomial computation analysis.** Suppose we want to evaluate the polynomial

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

at some point x_0.

(a) Consider the algorithm that computes first x_0^i for each i, using the fast exponentiation algorithm from lectures (without taking mod). Then, it computes $a_1 x_0$, $a_2 x_0^2$, $a_3 x_0^3, \ldots, a_n x_0^n$ (independently) and, finally, it adds all of these numbers to a_0 to obtain $p(x_0)$. How many sums and how many multiplications are involved in this algorithm? Please provide short explanations. (You can give your answer in O-notation.)

(b) We show next how to do better using *Horner's rule*. Prove that the following algorithm outputs $p(x_0)$.

Algorithm 1 Horner's rule

$z = a_n$
for $i = n-1$ `to` 0 **do**:
  $z = z \cdot x_0 + a_i$
end for
return z

(c) How many additions and multiplications does this algorithm use as a function of n? Please provide short explanations. (You can give your answer in O-notation.)

3. (20 pts.) **Exponentiation for large numbers.** Consider the following two algorithms for doing exponentiation.

Algorithm 2 NaiveExp

```
Input: An n-bit integer x and an m-bit integer y (assume n ≥ m).
z ← 1, i ← 0
while i < y do:
    z = z · x
    i = i + 1
end while
Output: z
```

Algorithm 3 FastExp

```
Input: An n-bit integer x and an m-bit integer y (assume n ≥ m).
if y == 0 then
    Output: 1
else
    z = FastExp(x, ⌊y/2⌋)
    if y is even then
        Output: z^2
    end if
    if y is odd then
        Output: x · z^2
    end if
end if
```

(a) Compute the running time of NaiveExp assuming that it takes $O(n \log n)$ to multiply an n-bit by an m-bit number provided $n \geq m$. (Hint: You may use the fact that $\sum_{k=1}^{L} k \log k = O(L^2 \log L)$.)

(b) Compute the running time of FastExp assuming that it takes $O(n \log n)$ to multiply an n-bit by an m-bit number provided $n \geq m$.

4. (20 pts.) **Fibonacci numbers.** There is an alternative way of computing Fibonacci numbers involving matrices. Note that we have:

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} F_{n-1} \\ F_n \end{pmatrix}.$$

If we write the latter equation recursively, we can get $\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$. Let $X = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$.

(a) Show that two 2×2 matrices can be multiplied using 4 additions and 8 multiplications.

(b) Show that for all $i < n$, all entries of X^i have $O(n)$ bits. (Hint: Consider the effect of each matrix multiplication on the bit count).

(c) The following recursive algorithm can be used to efficiently compute X^n.
Show that the running time of this algorithm is $O(M(n) \log n)$, where $M(n)$ is the time it takes to multiply two n-bit numbers. (Hint: first show that there are $O(\log n)$ recursive calls, and then show each call takes at most $O(M(n))$, you may use the results of parts (a) and (b) to show the latter).

Algorithm 4 `matrix`(X,n)

Input: X , n
if $n = 1$ **then**
  return X
end if
if n is even **then**
  $Z = \texttt{matrix}(X, \frac{n}{2})$
  return $Z \cdot Z$
end if
if n is odd **then**
  $Z = \texttt{matrix}(X, \frac{n-1}{2})$
  return $Z \cdot Z \cdot X$
end if
Output: `matrix(X,n)`

5. (20 pts.) **Heaps and Heap Sort.**

(a) What are the minimum and maximum numbers of nodes in a heap of height h?

(b) Is the array with values $\{10, 14, 19, 35, 31, 42, 27, 44, 26, 33\}$ a Min heap?

(c) Show that in the worst-case Heapify-UP could make $\Omega(\log n)$ swaps on a heap with n elements. (Hint: Give an example heap with n node values that would cause Heapify-UP to be called recursively at every node on a simple path up to the root).

6. (0 pts.) **Acknowledgments.** The assignment will receive a 0 if this question is not answered.
(a) If you worked in a group, list the members of the group. Otherwise, write "I did not work in a group."
(b) If you received significant ideas about the HW solutions from anyone not in your group, list their names here. Otherwise, write "I did not consult with anyone other than my group members."
(c) List any resources besides the course material that you consulted in order to solve the material. If you did not consult anything, write "I did not consult any non-class materials."